Leveraging Program Analysis for Incremental Computation over Dynamically Changing Datasets in IPython Notebooks

Konstantinos Xirogiannopoulos and Benjamin Bengfort

Department of Computer Science
University of Maryland
{kostasx,bengfort}@cs.umd.edu

October 16, 2015

1 Proposal

As advanced data analysis techniques have become mainstream, programmers are easily able to apply complex data analysis and computation on diverse datasets, thereby enabling organizations to have a better understanding of their information and enhance their decision making processes. Specific programming languages like R and Python are being widely adopted and in particular, and tools like `IPython Notebooks` [2] are being used for interactive data analytics and visualization. This has caused a significant shift – it is no longer the application or tool that is primarily distributed in this context, but rather the combination of code and data in notebook form.

IPython Notebooks offer a flexible means of interactive computation by mixing input datasets, analytical Python code, explanation, and output reports all in a single presentation format. Because the notebooks are saved in a JSON format and rendered on demand, they can be versioned, cloned and modified. However, in an age of constant data influx and continuously changing datasets - IPython views are doomed to a constant stale state in the environment in which they are executed. Changes in the underlying data require users to re-evaluate their queries and computations by executing the notebook code across the entire dataset. Instead, we propose a system that utilizes program analysis techniques to apply arbitrary incremental computation on the changed portion of the data and output without rerunning the analysis on the entirety of the modified dataset.

Program analysis is a powerful tool for automatically reasoning about the structure of programs by studying the approximations of program properties. We believe that program analysis can be used to discover program properties in advance of run-time to allow for optimization and correction. In particular, programs can be written in an incremental fashion by reasoning about first-class *names* since they are the primary identifiers of the location of incremental computations [1]. Additionally, there has been interest on conducting program analysis on IPython Notebooks in order to extract provenance information [3] such that users can backtrack commands and actions to fix errors as the environment changes.

Motivated by the above work, we propose to use program analysis to study the incremental re-evaluation of IPython notebooks that read from dynamically changing datasets. We believe that by understanding the program structure and building links between variables and data points in a constantly changing environment, we can track potential changes and ultimately update a notebook view efficiently without a complete recomputation.

References

[1] Matthew A Hammer, Joshua Dunfield, Kyle Headley, Nicholas Labich, Jeffrey S Foster, Michael Hicks, and David Van Horn. Incremental computation with names. *arXiv preprint arXiv:1503.07792*, 2015.

[2] Fernando Pérez and Brian E Granger. Ipython: a system for interactive scientific computing. *Computing in Science & Engineering*, 9(3):21–29, 2007.

[3] João Felipe Nicolaci Pimentel, Vanessa Braganholo, Leonardo Murta, and Juliana Freire. Collecting and analyzing provenance on interactive notebooks: when ipython meets noworkflow. In *Proceedings of the 7th USENIX Conference on Theory and Practice of Provenance*, pages 10–10. USENIX Association, 2015.